What is claimed is:

1. A digital computer including at least one processor producing virtual addresses over a range of virtual addresses, a translation buffer coupled to said at least one processor for translating the virtual addresses to physical addresses, and a random access memory coupled to the translation buffer for addressing by the physical addresses and coupled to said at least one processor for supplying data to said at least one processor, wherein the random access memory contains physical memory having a range of physical addresses that is greater than the range of virtual addresses, wherein the digital computer is programmed with a plurality of virtual-to-physical address mappings to define a plurality of virtual memory spaces, each of the plurality of virtual memory spaces includes common physical memory that is included in the other of the virtual memory spaces, and at least one of the virtual memory spaces includes a chunk of physical memory that is not included in any other of the plurality of virtual memory spaces, the chunk of physical memory that is not included in any other of the plurality of virtual memory spaces being assigned for use by a software module, and the digital computer being programmed for using the common physical memory for communication of parameters to and results from the software module.

2. The digital computer as claimed in claim 1, wherein the common physical memory includes physical memory allocated to a stack for said at least one processor.

3. The digital computer as claimed in claim 1, wherein each of the virtual memory spaces includes a chunk of physical memory allocated to BIOS and device drivers, the chunk of physical memory allocated to BIOS and device drivers being common to the plurality of virtual memory spaces.

4. The digital computer as claimed in claim 1, wherein at least one other of the virtual memory spaces is directly mapped to a bottom region of the physical memory address space and is allocated to page tables.

5. The digital computer as claimed in claim 1, wherein the digital computer is programmed for copying at least one parameter from a context of an application to the common physical memory, switching virtual address translation from a virtual memory space of the application to said at least one of the virtual memory spaces, executing the software module for processing said at least one parameter to produce a result placed in the common physical memory, switching the virtual address translation back to the virtual memory space of the application, and copying the result from the common physical memory to the context of the application.

6. The digital computer as claimed in claim 5, wherein the virtual memory space of the application is directly mapped to a bottom region of the physical memory address space, the digital computer is programmed for switching virtual address translation from the virtual memory space of the application to said at least one of the virtual memory spaces by turning paging on, and the digital computer is programmed for switching

virtual address translation from said at least one of the virtual memory spaces to the virtual memory space of the application by turning paging off.

7. The digital computer as claimed in claim 1, wherein the digital computer is programmed for switching virtual address translation from the virtual memory space of an application to said at least one of the virtual memory spaces and executing the software module by disabling thread scheduler preemption of the current thread, copying at least one parameter from a context of the application to the common physical memory, switching virtual address translation from the virtual memory space of the application to said at least one of the virtual memory spaces, executing the software module for processing said at least one parameter to produce a result placed in the common physical memory, switching the virtual address translation back to the virtual memory space of the application, copying the result from the common physical memory to the context of the application, and enabling thread scheduler preemption of the current thread.

8. The digital computer as claimed in claim 1, wherein the digital computer is programmed for moving data between network clients and data storage, and the plurality of virtual address spaces include at least a first virtual address space containing an inode cache, a second virtual address space containing a dynamic name lookup cache for finding an inode having a given path name, and a third virtual address space containing a block map for snapshot copies.

9. A digital computer including at least one processor producing virtual addresses over a range of virtual addresses, a translation buffer coupled to said at least one processor for translating the virtual addresses to physical addresses, and a random access memory coupled to the translation buffer for addressing by the physical addresses and coupled to said at least one processor for supplying data to said at least one processor, wherein the random access memory contains physical memory having a range of physical addresses that is greater than the range of virtual addresses, wherein the digital computer is programmed with a plurality of virtual-to-physical address mappings to define a plurality of virtual memory spaces, each of the plurality of virtual memory spaces includes common physical memory that is included in the other of the virtual memory spaces, and each of the plurality of virtual memory spaces including at least one respective separate chunk of physical memory that is not included in any other of the virtual memory spaces, each of the respective separate chunks of physical memory being assigned for use by a respective software module, the digital computer being programmed for using the common physical memory for communication of parameters to and results from the software module, and the plurality of virtual memory spaces including at least a first virtual memory space that is directly mapped to a bottom region of the physical memory address space, a second virtual memory space, and a third virtual memory space.

10. The digital computer as claimed in claim 9, wherein the common physical memory is at the bottom of the physical address space and includes memory allocated to

at least one processor stack, and the respective software module assigned to the separate chunk of physical memory in the first virtual address space accesses buffer cache.

11. The digital computer as claimed in claim 9, wherein each of the virtual memory spaces includes a chunk of physical memory allocated to BIOS and device drivers, and the chunk of physical memory allocated to BIOS and device drivers is included in each of the plurality of virtual memory spaces and is mapped to a top region of each of the plurality of virtual memory spaces.

12. The digital computer as claimed in claim 9, wherein the digital computer is programmed for moving data between network clients and data storage, and the respective software modules include software for accessing a dynamic name lookup cache in the separate chunk of physical memory in the second virtual address space, and snapshot copy software for accessing at least one block map in the separate chunk of physical memory in the third virtual address space.

13. A digital computer including at least one processor producing virtual addresses over a range of virtual addresses, a translation buffer coupled to said at least one processor for translating the virtual addresses to physical addresses, and a random access memory coupled to the translation buffer for addressing by the physical addresses and coupled to said at least one processor for supplying data to said at least one processor, wherein the random access memory contains physical memory having a range of physical addresses that is greater than the range of virtual addresses, wherein the digital computer

is programmed with a plurality of virtual-to-physical address mappings to define a plurality of virtual memory spaces, each of the plurality of virtual memory spaces includes at least one common chunk of physical memory that is included in the other of the virtual memory spaces, each of the plurality of virtual memory spaces includes at least one respective separate chunk of physical memory that is not included in any other of the virtual memory spaces, each of the respective separate chunks of physical memory is assigned for use by a respective software module, the digital computer is programmed for using the common chunk of physical memory for communication of parameters to and results from the software module, and the plurality of virtual memory spaces includes at least a first virtual memory space that is directly mapped to a bottom region of the physical memory address space, a second virtual memory space, and a third virtual memory space;

wherein the common chunk of physical memory is at the bottom of the physical address space and includes memory allocated to at least one processor stack, and the respective software module assigned to the separate chunk of physical memory in the first virtual address space includes accesses a buffer cache; and

wherein each of the virtual memory spaces includes a chunk of physical memory allocated to BIOS and device drivers, and the chunk of physical memory allocated to BIOS and device drivers is included in each of the plurality of virtual memory spaces and is mapped to a top region of each of the plurality of virtual memory spaces.

14. The digital computer as claimed in claim 13, wherein the digital computer is programmed for moving data between network clients and data storage, and the

respective software modules include software for accessing a dynamic name lookup cache in the separate chunk of physical memory in the second virtual address space, and snapshot copy software for accessing at least one block map in the separate chunk of physical memory in the third virtual address space.

15. A method of operating a digital computer for executing a first software module and a second software module, the first software module accessing a first virtual memory space and the second software module accessing a second virtual memory space, each of the first and second virtual memory spaces containing common physical memory, the first virtual memory space including a first separate chunk of physical memory that is not included in the second virtual memory space and that is accessed by the first software module, the second virtual memory space including a second separate chunk of physical memory that is not included in the first virtual memory space and that is accessed by the second software module, wherein the method includes transferring execution between the first software module and the second software module by:

executing the first software module to place at least one parameter in the common physical memory;

switching virtual-to-physical address translation from the first virtual memory space to the second virtual memory space;

executing the second software module to produce a result from said at least one parameter obtained from the common physical memory, the result being placed in the common physical memory,

switching virtual-to-physical address translation from the second virtual memory space to the first virtual memory space; and

executing the first software module to obtain the result from the common physical memory.

16. The method as claimed in claim 15, wherein the switching of the virtual-to-physical address translation from the first virtual memory space to the second virtual memory space is performed by turning paging on, and the switching of the virtual-to-physical address translation from the second virtual memory space to the first virtual memory space is performed by turning paging off.

17. The method as claimed in claim 15, which includes an additional initial step of turning thread scheduler preemption off, and an additional final step of turning the thread scheduler preemption on.

18. The method as claimed in claim 15, which includes addressing more memory space in physical memory than can be addressed by the processor in any one of the first virtual address space and the second virtual address space.

19. The method as claimed in claim 15, which includes the digital computer moving data between network clients and data storage, and execution of the second software module accesses a dynamic name lookup cache in the separate chunk of physical memory contained in the second virtual address space.

20. The method as claimed in claim 19, which includes the digital computer executing snapshot copy software to access a block map in a third virtual address space.